МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**КУРСОВАЯ РАБОТА**

**ИССЛЕДОВАНИЕ МЕТОДОВ БИНАРИЗАЦИИ ИЗОБРАЖЕНИЙ МЕДИЦИНСКИХ ЛАБОРАТОРНЫХ АНАЛИЗОВ**

Работу выполнила \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.С. Агаджанян

(подпись)

Направление подготовки 02.03.02 — «Фундаментальная информатика и\_\_\_\_\_

(код, наименование)

информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_курс\_\_\_\_3\_\_\_\_\_

Направленность (профиль) \_\_\_\_\_\_\_\_\_\_\_\_Математическое и программное обеспечение компьютерных технологий\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Научный руководитель

канд. техн. наук, доц. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Т. А. Приходько

(подпись, дата)

Нормоконтролер

преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Е. А. Нигодин

(подпись, дата)

Краснодар

2024

**РЕФЕРАТ**

Курсовая работа 29 стр., 4 ч., 17 рис., 4 источ., 1 приложение.

КОМПЬЮТЕРНОЕ ЗРЕНИЕ, БИНАРИЗАЦИЯ ИЗОБРАЖЕНИЙ, PYTHON, ПРИЛОЖЕНИЕ

Цель работы – разработать приложение, позволяющее с помощью различных методов бинаризировать изображения и оценить качество бинаризации.

Объектом исследования в работе являются методы бинаризации изображений медицинских лабораторных анализов.

Предметом исследования являются алгоритмы бинаризации изображений.

В качестве методов исследования использовались, математическое и программное моделирование, сравнительный анализ. В результате работы были изучены методы Ниблэка, Бернсена, Кристиана, а также методы сравнения качества: среднеквадратическая ошибка.

Научная новизна работы заключается в исследовании методов бинаризации изображений и разработки приложения для их сравнения.

По результатам исследования, было разработано приложение, которое осуществляет бинаризацию изображения с помощью трех различных методов и выполняет оценку качества бинаризации.

**СОДЕРЖАНИЕ**

[Введение](#_Toc121868742) 4

[1 Теоретическая информация о методах бинаризации](#_Toc121868743) 7

[1.1 Понятие бинаризации](#_Toc121868744) 7

[1.1.1 Бинаризация с нижним порогом](#_Toc121868760) 8

[1.1.2 Бинаризация с верхним порогом](#_Toc121868760) 9

[1.1.3 Бинаризация с двойным ограничением](#_Toc121868760) 9

[1.1.4 Неполная пороговая обработка](#_Toc121868760) 9

[1.1.5 Многоуровневое пороговое преобразование](#_Toc121868760) 10

[1.1.6 Локальные (адаптивные) методы бинаризации](#_Toc121868760) 11

[1.2 Метод Ниблэка.](#_Toc121868745) 12

[1.3 Метод Бернсена.](#_Toc121868746) 13

[1.4 Метод Кристиана](#_Toc121868747) 14

[2 Методы оценки качества бинаризации изображений](#_Toc121868749) 15

[2.1 Среднеквадратическая ошибка (MSE)](#_Toc121868750) 15

[3 Программная реализация](#_Toc121868752) 18

[3.1 Инструменты для разработки](#_Toc121868753) 18

[3.2 Требования к реализации](#_Toc121868754) 18

[3.3 Описание функциональности приложения](#_Toc121868755) 19

[4 Исследовательская часть](#_Toc121868756) 22

[4.1 Результаты работы приложения. Пример 1](#_Toc121868757) 22

[4.2 Результаты работы приложения. Пример 2](#_Toc121868758) 23

[4.3 Результаты работы приложения. Пример 3](#_Toc121868759) 25

[4.4 Результаты работы приложения. Пример 4](#_Toc121868760) 26

[4.5 Оценка полученных результатов](#_Toc121868760) 27

[Заключение](#_Toc121868761) 28

[Список использованных источников](#_Toc121868762) 29

[Приложение А Основная программа](#_Toc121868763) 30

**ВВЕДЕНИЕ**

Одной из ключевых задач при исследовании фотоснимков лабораторных медицинских анализов является выделение объектов интереса (ОИ). Это критически важный этап, поскольку он позволяет не только идентифицировать и количественно оценить различные структуры, но и проводить более глубокий анализ их морфологических характеристик. В контексте медицинской диагностики, таких как анализы крови или другие биоматериалы, правильное выделение ОИ может существенно повлиять на точность диагностики и последующее лечение пациентов.

Выделение ОИ включает в себя несколько этапов обработки изображений, и одним из первых шагов является бинаризация. Бинаризация — это процесс преобразования цветного или градационного изображения в черно-белое, где каждый пиксел получает одно из двух значений: 0 или 1. Значение 0 соответствует пикселу черного цвета, а значение 1 — пикселу белого цвета. Этот процесс позволяет упростить изображение и сосредоточиться на тех элементах, которые представляют собой объекты интереса.

Бинаризация играет важную роль в дальнейшей обработке изображений. После того как изображение было бинаризовано, можно применять различные алгоритмы для анализа и подсчета объектов. Например, после выделения белых пикселей, представляющих ОИ, можно легко определить их количество, провести анализ формы и размера, а также выявить другие характеристики, такие как распределение и плотность объектов.  
Кроме того, бинаризация помогает устранить шум и ненужные детали, которые могут затруднить дальнейший анализ. Это особенно важно в медицинских приложениях, где точность и надежность данных имеют первостепенное значение.

Таким образом, выделение объектов интереса через бинаризацию является основополагающим шагом в обработке фотоснимков лабораторных медицинских анализов. Этот процесс не только упрощает дальнейший анализ, но и обеспечивает более высокую точность в диагностике и исследовании различных медицинских состояний. Правильная реализация бинаризации и последующей обработки изображений может существенно повысить качество медицинских исследований и улучшить результаты для пациентов.

Актуальность данной работы обуславливается:

1. Рост объема медицинских данных**.** С увеличением объемов данных, получаемых в результате лабораторных анализов, возникает необходимость в автоматизации обработки изображений для повышения эффективности диагностики.
2. Увеличение точности диагностики**.** Правильная бинаризация изображений позволяет более точно выделять объекты интереса, что напрямую влияет на качество диагностики и лечение пациентов.
3. Развитие технологий. Современные технологии обработки изображений и алгоритмы машинного обучения открывают новые возможности для улучшения методов бинаризации, что делает тему особенно актуальной.
4. Потребность в стандартизации. В условиях разнообразия методов и технологий, используемых в медицинской практике, необходимо разработать стандартизированные подходы к бинаризации, чтобы обеспечить сопоставимость результатов.
5. Поддержка клинических решений. Эффективные методы бинаризации могут стать основой для разработки систем поддержки принятия клинических решений, что позволит врачам быстрее и точнее интерпретировать результаты анализов.

Основная цель работы – познакомиться с тремя методами бинаризации изображений: метод Ниблэка, метод Бернсена, метод Кристиана и сравнить результаты обработки.

Для реализации поставленной цели предполагается решить следующие задачи:

* изучить принципы работы данных методов;
* выбрать, изучить и реализовать методы сравнения полученных изображений;
* выбрать инструментарий для разработки приложения и спроектировать его;
* разработать методические рекомендации по использованию приложения.

Объектом исследования в работе являются изображения медицинских лабораторных анализов, полученные с помощью различных методов визуализации, таких как микроскопия, цифровая фотосъемка и другие технологии.

Предметом исследования являются алгоритмы бинаризации изображений, применяемые для выделения объектов интереса в медицинских анализах.

Информационная база исследования включает в себя несколько видов учебных материалов таких как книги с обзорами методов бинаризации, статьи от авторитетных сервисов и документация. В качестве методов исследования использовались абстрагирование, наблюдение, моделирование и эксперименты.

Научная новизна работы заключается в том, что предлагается анализ алгоритмов на основе двух методов оценки качества.

Теоретическая и практическая значимость работы состоит в результатах исследования, которые могут быть использованы студентами при изучении методов бинаризации. Результаты работы также могут быть внедрены в робототехнические комплексы, так как не требуют большой вычислительной мощности.

Исследуемые алгоритмы используются в области обработки изображений.

1. **Теоретическая информация о методах бинаризации**
   1. **Понятие бинаризации**

Процесс бинаризации представляет собой перевод цветного изображения или изображения в градациях серого в двухцветное черно-белое.

Целью операции бинаризации является радикальное уменьшение количества информации, содержащейся на изображении.

Главным параметром такого преобразования является порог — значение, с которым сравнивается яркость каждого пикселя. После сравнения, пикселю присваивается одно из двух возможных значений: 0 - «граница объекта» или 1 - «остальная область» [2].

Общая схема бинаризации представлена на рисунке 1.



Бинаризация используется для распознавания объектов, форм и, в частности, символов. Благодаря бинаризации можно отличить интересующий объект от фона, на котором он находится.

Существуют различные методы бинаризации, которые можно условно разделить на две группы:

1. глобальные (пороговые);
2. локальные (адаптивные).

В глобальных методах бинаризации происходит работа со всем изображением сразу. В ходе работы находится порог бинаризации t, с помощью которого происходит деление на черное и белое, причем величина порога t остается неизменной в течение всего процесса бинаризации.

К пороговым методам бинаризации относятся:

1. бинаризация с нижним порогом;
2. бинаризации с верхним порогом;
3. бинаризация с двойным ограничением;
4. неполная пороговая обработка;
5. многоуровневое пороговое преобразование
   * 1. **Бинаризация с нижним порогом**

Бинаризация с нижним порогом является наиболее простой операцией, в которой используется только одно значение порога.

Суть метода заключается в том, что значение амплитуд пикселей, которые входят в диапазон [a,b] — становятся черными, которые не входят белым.

Реализовывается согласно правилу:

где  *–* яркость пикселя на исходном изображении,  ,

*–* значение пикселя результирующего изображения, ,

*t* – порог бинаризации.

Данная операция преобразует полутоновое изображение в бинарное. Любая точка изображения, для которой выполняется условие

*f (m, n)* > *t* называется точкой объекта, а в противном случае – точкой фона.

* + 1. **Бинаризации с верхним порогом**

Бинаризации с верхним порогом – это вариант метода бинаризации с нижним порогом, в результате которого получается негатив исходного изображения.

Операция бинаризации с верхним порогом:

* + 1. **Бинаризация с двойным ограничением**

Бинаризация с двойным ограничением используется в случае, если необходимо выделить определенные области, значения яркости пикселей в которых могут изменяться в определенном диапазоне.

Формула бинаризации с двойным ограничением:

где *t1* и *t2* – пороги бинаризации, причем *t1* < *t2*.

* + 1. **Неполная пороговая обработка**

Неполная пороговая обработка используется в случае, если необходимо получить наиболее простое для дальнейшего анализа изображение, в ходе которого изображение лишается фона со всеми его деталями, которые были на исходном изображении.

Формула неполной пороговой бинаризации:

* + 1. **Многоуровневое пороговое преобразование**

Многоуровневое пороговое преобразование формирует изображение, не являющееся бинарным, но состоящее из сегментов с различной яркостью. Однако, при этом, полученное в ходе преобразования изображение уже не будет являться бинарным.

Формула данного преобразования:

где f(m, n) – входное изображение (значение яркости пикселя);

f ‘(m, n) – выходное изображение (значение яркости пикселя);

t– порог (параметр обработки);

D1, D2,…,Dn - области изображения.

Если значение порога *t* является одинаковым для всех элементов изображения *f(m,n)*, то такой порог называется глобальным. Если значение порога подбирается различным в разных частях изображения, то он называется локальным. Если к тому же порог *t* зависит от пространственных координат, то он называется динамическим. Это можно выразить формулой:

где *t (m, n) –* порог, зависящий от координат *m* и *n*.

В случае же, если порог зависит от некоторой локальной характеристики пикселя *(m, n)* изображения, например, от средней яркости в окрестности с центром в этой точке, то он называется адаптивным.

* + 1. **Локальные (адаптивные) методы бинаризации**

Адаптивная бинаризация – это метод обработки изображений, который

позволяет определить пороговое значение для каждой области изображения отдельно, в зависимости от особенностей освещения и контрастности в этой области. Это позволяет получить более точный результат по сравнению с пороговой бинаризацией, где одно и то же пороговое значение применяется для всего изображения [1].

Процесс адаптивной бинаризации состоит из следующих шагов:

1. Разбиение изображения на небольшие неперекрывающиеся блоки.
2. Расчет порогового значения для каждого блока. Обычно для этого используются методы вычисления статистических параметров яркости пикселей в блоке, таких как среднее значение или медиана.
3. Рассмотрение каждого пикселя изображения и сравнение его яркости с пороговым значением блока, к которому он принадлежит.
4. Если яркость пикселя выше порогового значения блока, то ему присваивается белый цвет, а если ниже – черный цвет.

Одним из главных преимуществ адаптивной бинаризации является возможность учитывать неоднородность освещения на изображении, что обычно является проблемой для пороговой бинаризации.

Рассмотрим наиболее популярные адаптивные методы бинаризации изображений:

1. метод Ниблэка;
2. метод Бернсена;
3. метод Кристиана.
   1. **Метод Ниблэка**

Идея данного метода состоит в варьировании порога яркости в бинаризации от точки к точке на основании локального значения стандартного отклонения. Порог яркости в точке (x, y) рассчитывается по следующей формуле:

где μ(x, y) – среднее и s(x, y) — среднеквадратичное отклонение выборки для некоторой окрестности точки. Размер окрестности должен быть минимальным, но таким, чтобы сохранить локальные детали изображения. В то же время размер должен быть достаточно большим, чтобы понизить влияние шума на результат;

k определяет, какую часть границы объекта взять в качестве самого объекта. Значение k=-0.2 задает достаточно хорошее разделение объектов, если они представлены черным цветом, а значение k=+0.2, – если объекты представлены белым цветом.

Метод Ниблэка за счет своей простоты позволяет достичь наиболее высокую скорость бинаризации изображений. Метод используется на практике для быстрой фильтрации контрастных изображений, на которых практически отсутствуют сильно зашумленные участки с плавными переходами яркости.

Достоинством данного метода являются быстрая скорость работы алгоритма, в ходе которых получаются контрастные изображения.

Основным недостатком этого метода является сильная зашумленность бинарного изображения. Это значительно затрудняет обработку изображения на последующих этапах.

**1.3 Метод Бернсена**

Метод Бернсена – один из самых распространённых методов локального определения порога бинаризации. Используется для схематических и картографических изображений.

Метод Бернсена предполагает деление всего изображения на квадраты r×r , где r – нечетное число, с центром в точке (m, n). Для каждого пикселя изображения в пределах квадрата используется порог, имеющий значение

где jhigh – наибольший уровень яркости в квадрате, jlow - наименьший уровень яркости в квадрате.

Если в принятой области используемая мера контраста удовлетворяет условию

G(m,n) = (jhigh − jlow) ≤ ε, где ε - заданная пороговая величина, то исследуемый квадрат содержит объекты только одного класса: объектов или фона.

Если текущий пиксель t(m,n) < e, где e представляет собой константу, то пиксель будет относиться только к одному из видов: черному или белому. Если среднее отклонение меньше порога контраста, то рассматриваемый пиксель становится того цвета, который задавался параметром «цвет сомнительного пикселя».

Достоинством описанного метода является его сравнительная простота и скорость работы. Основным недостатком является возникновение сильных помех при обработке больших монотонных областей, что в ряде случаев приводит к появлению ложных черных пятен.

**1.4 Метод Кристиана**

Суть метод Кристиана заключается в том, чтобы определить местное пороговое значение, нормализуя контраст и шум изображения следующим образом:

где k = 0.5

M – минимальное серое значение всего изображения

R - максимальное среднеквадратичное отклонение серого значения из локального окна.

Достоинством данного метода является то, что достигаются одни из лучших результатов бинаризации среди методов локального значения, описанные выше. Также метод позволяет работать с изображениями, у которых мало освещения, особенно в случаях, когда значения пикселей объекта находятся близко друг к другу. Однако, производительность метода Кристина значительно ухудшается, если имеются заметные изменения серого фоновых значений по всему изображению.

**2 Методы оценки качества сегментирования изображений**

**2.1 Среднеквадратическая ошибка (MSE)**

В большинстве обучающих сетей ошибка рассчитывается как разница между фактическим выходным значением y и прогнозируемым выходным значением ŷ. Функция, используемая для вычисления этой ошибки, известна как функция потерь, также часто называемая функцией ошибки или затрат.

Рассмотрим следующую функцию потерь, называемую MSE.

Средняя квадратичная ошибка (MSE – с англ. mean squared error) — это распространённая функция потерь, используемая в машинном обучении.

Среднеквадратичная ошибка широко используется во многих приложениях машинного обучения, включая:

* Задачи регрессии, где требуется предсказать непрерывное значение на основе входных данных.
* Задачи прогнозирования временных рядов, где требуется предсказать будущие значения на основе прошлых и текущих данных.
* Любой другой тип задачи машинного обучения, который включает предсказание непрерывного значения.

Целями вычисления значения MSE являются:

1. Оценка производительности  —  визуальное определение того, насколько хорошо работает модель. Другими словами, это возможность быстро понять, с чем предстоит работать.
2. Оптимизация модели позволяет выяснить, достигнуто ли наилучшее из возможных соответствий или же требуются улучшения. Другими словами, определить, какая модель максимально подходит для работы с выбранными точками данных.

К преимуществам среднеквадратичной ошибки относятся:

* Интерпретация: Среднеквадратичная ошибка предоставляет прямую интерпретацию. Она измеряет среднее квадратичное отклонение между истинными и предсказанными значениями.
* Чувствительность к выбросам: Среднеквадратичная ошибка чувствительна к выбросам, поскольку она штрафует модель за большие ошибки.

Среднеквадратичная ошибка (MSE) рассчитывается как среднее значение квадратов разностей между прогнозируемыми и фактически наблюдаемыми значениями. Математически это можно выразить следующим образом:

где  *n* — количество наблюдений по которым строится модель и количество прогнозов, - фактические значение [зависимой переменной](https://wiki.loginom.ru/articles/output-variable.html) для *i*-го наблюдения, - значение зависимой переменной, предсказанное моделью.

Эта формула вычисляет среднее значение квадратов ошибок . Потери высоки, когда предсказания модели сильно отличаются от истинных значений, и низки, когда предсказания модели соответствуют истинным значениям. Это побуждает модель делать точные предсказания.

В матрице формула выглядит следующим образом:

Недостатком использования MSE является то, что если на одном или нескольких неудачных примерах, возможно, содержащих [аномальные значения](https://wiki.loginom.ru/articles/outlier.html) будет допущена значительная ошибка, то возведение в квадрат приведёт к ложному выводу, что вся модель работает плохо. Аномальные значения – это значения, которые не укладываются в общую модель поведения анализируемого процесса. Они сильно отличаются от окружающих данных и могут быть вызваны как ошибками измерений, так и некорректным вводом данных

С другой стороны, если модель даст небольшие ошибки на большом количестве [примеров](https://wiki.loginom.ru/articles/training-sample.html), то может возникнуть обратный эффект — недооценка слабости модели.

**3 Программная реализация**

Практическая часть исследования состоит в написании приложения, которое позволит наглядно показать работу методов и сравнить их. Код размещен на веб-сервисе контроля версий GitHub. На рисунке 2 изображено наполнение проекта.

Изображение выглядит как текст, Шрифт, линия, число

Автоматически созданное описание

Рисунок 2 – Наполнение проекта на GitHub

**3.1 Инструменты для разработки**

Для разработки был использован язык Python 3.12, графическая библиотека Tkinter [5] для создания программы с оконным интерфейсом. Также были использованы:

* библиотека PIL для открытия и работы с изображениями разных форматов;
* библиотека OpenCV для обработки изображений;
* модуль NumPy для манипуляции с большими массивами и матрицами;
* библиотека time для работы со временем.

**3.2 Требования к реализации**

Приложение должно содержать:

* удобный и понятный интерфейс;
* инструкцию по эксплуатации;
* иметь функционал для сохранения результатов обработки.

**3.3 Описание функциональности приложения**

На рисунке 3 изображен интерфейс приложения. Выпадающее меню в верхнем левом углу экрана является важным элементом пользовательского интерфейса, обеспечивающим доступ к основным функциям приложения. Его расположение, визуальное представление и функциональность способствуют удобной и интуитивно понятной навигации.

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 3 - Начальный экран приложения

После выбора изображения для обработки на экране высветятся результаты работы метода Ниблэка, Бернсена и Кристиана. Пример показан на рисунке 4.

Изображение выглядит как текст, снимок экрана, шаблон

Автоматически созданное описание

Рисунок 4 – Экран приложения после выбора изображения для обработки

В разделе «О программе» содержится информация о названии, версии и авторе приложения.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 5 – Окно раздела «О программе»

В разделе «Помощь» содержится информация о том, как пользоваться данной программе.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 6 – Окно раздела «Помощь»

При нажатии на «Файл» откроется выпадающее меню, в котором можно выбрать изображение для обработки, после чего его можно удалить из программы или сохранить результаты обработки.

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 7 – Выпадающее меню раздела «Файл»

1. **Исследовательская часть**

Рассмотрим работу приложения на разных изображениях и сравним результаты обработки.

**4.1 Результаты работы приложения. Пример 1**

На рисунке 10 представлено исходное, зашумленное изображение общего анализа крови. Рисунок 11 демонстрирует, как это изображение было преобразовано разработанным приложением, представляя собой результаты его работы.

**Изображение выглядит как растение, грибок

Автоматически созданное описание со средним доверительным уровнем**

Рисунок 10 – Входные данные

|  |  |  |
| --- | --- | --- |
| Изображение выглядит как текст, шаблон, ткань, искусство  Автоматически созданное описание | Изображение выглядит как текст, круг, снимок экрана, искусство  Автоматически созданное описание | Изображение выглядит как зарисовка, рисунок, дизайн, шаблон  Автоматически созданное описание |
| Рисунок 11 – Результаты работы алгоритмов  а) Ниблэка б) Бернсена с) Кристиана | | |

В таблице 1 представлена сравнительная характеристика методов Ниблэка, Бернсена, Кристиана.

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Рассматриваемый метод | Величина MSE | Время выполнения (в секундах) |
| 1 | Метод Ниблэка | 15398,751416 | 0,8233297 |
| 2 | Метод Бернсена | 4685,7383165 | 0,3559887 |
| 3 | Метод Кристиана | 17300,7862896 | 0,0002377 |

**4.2 Результаты работы приложения. Пример 2**

На рисунке 12 представлено исходное изображение шпорообразных спирилл. Рисунок 13 демонстрирует, как это изображение было преобразовано разработанным приложением, представляя собой результаты его работы.

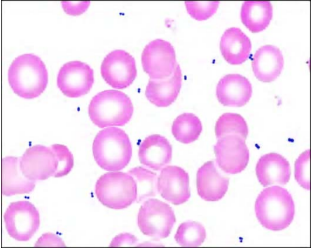


Рисунок 12 – Входные данные

|  |  |  |
| --- | --- | --- |
|  | Изображение выглядит как текст, снимок экрана, плакат, круг  Автоматически созданное описание | Изображение выглядит как текст, Шрифт, снимок экрана, Графика  Автоматически созданное описание |
| Рисунок 13 – Результаты работы алгоритмов  а) Ниблэка б) Бернсена с) Кристиана | | |

В таблице 2 представлена сравнительная характеристика методов Ниблэка, Бернсена, Кристиана.

Таблица 2

|  |  |  |  |
| --- | --- | --- | --- |
| № | Рассматриваемый метод | Величина MSE | Время выполнения (в секундах) |
| 1 | Метод Ниблэка | 16343,220559 | 0,9538741 |
| 2 | Метод Бернсена | 4685,7021815 | 0,4062587 |
| 3 | Метод Кристиана | 6888,50301126 | 0,0001995 |

**4.3 Результаты работы приложения. Пример 3**

На рисунке 14 представлено исходное изображение бактерий стрептококка пневмонии. Рисунок 15 демонстрирует, как это изображение было преобразовано разработанным приложением, представляя собой результаты его работы.

Изображение выглядит как червь, зарисовка, беспозвоночный, черно-белый

Автоматически созданное описание

Рисунок 14 – Входные данные

|  |  |  |
| --- | --- | --- |
| Изображение выглядит как текст, одежда, ткань, шаблон  Автоматически созданное описание |  | Изображение выглядит как зарисовка, рисунок, червь, иллюстрация  Автоматически созданное описание |
| Рисунок 15 – Результаты работы алгоритмов  а) Ниблэка б) Бернсена с) Кристиана | | |

В таблице 3 представлена сравнительная характеристика методов Ниблэка, Бернсена, Кристиана.

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| № | Рассматриваемый метод | Величина MSE | Время выполнения (в секундах) |
| 1 | Метод Ниблэка | 13623,5775014 | 0,9771119 |
| 2 | Метод Бернсена | 8680,402479 | 0,3913344 |
| 3 | Метод Кристиана | 3671,59547031 | 0,00024 |

* 1. **Результаты работы приложения. Пример 4**

На рисунке 16 мы видим исходное изображение нейронов Пуркинье. Рисунок 17 демонстрирует результат применения разработанного приложения для бинаризации этого изображения, показывая состояние после обработки.

Изображение выглядит как Красочность, фиолетовый, Фиолетовый, Сирень

Автоматически созданное описание

Рисунок 16 – Входные данные

|  |  |  |
| --- | --- | --- |
| Изображение выглядит как текст, Шрифт, шаблон, плакат  Автоматически созданное описание | Изображение выглядит как текст, снимок экрана, плакат, графический дизайн  Автоматически созданное описание | Изображение выглядит как текст, снимок экрана, зарисовка, шаблон  Автоматически созданное описание |
| Рисунок 17 – Результаты работы алгоритмов  а) Ниблэка б) Бернсена с) Кристиана | | |

В таблице 4 представлена сравнительная характеристика методов Ниблэка, Бернсена, Кристиана.

Таблица 4

|  |  |  |  |
| --- | --- | --- | --- |
| № | Рассматриваемый метод | Величина MSE | Время выполнения (в секундах) |
| 1 | Метод Ниблэка | 19321,433564 | 0,9604486 |
| 2 | Метод Бернсена | 16641,063262 | 0,4038403 |
| 3 | Метод Кристиана | 6892,79284315 | 0,0002376 |

**4.5 Оценка полученных результатов**

Из предложенных результатов работы методов бинаризации изображений медицинских лабораторных анализов согласно приведенным методам оценки качества бинаризации можно заметить, что для данных изображений самым эффективным является метод Кристиана. Вторым по эффективности можно соотнести метод Бернсена. Замыкающим же является метод Ниблэка.

**ЗАКЛЮЧЕНИЕ**

В современном мире обработки изображений бинаризация играет ключевую роль в упрощении анализа визуальных данных. В ходе работы были изучены различные методы бинаризации изображений. В процессе исследования были подробно изучены возможности языка программирования Python и его библиотек, таких как OpenCV, Pillow и NumPy. Эти инструменты позволяют эффективно обрабатывать изображения, реализовывать методы бинаризации и визуализировать результаты. Python стал выбором благодаря своей простоте и мощным возможностям для анализа данных.

По результатам исследования было разработано приложение, которое позволяет наглядно рассмотреть результаты работы различных методов бинаризации на различных изображениях. Приложение предоставляет пользователю возможность загружать изображения, выбирать методы бинаризации и сравнивать результаты в удобном интерфейсе. Это делает процесс обучения и анализа более интерактивным и доступным.

В дальнейшем планируется углубленное изучение работы в области компьютерного зрения, с акцентом на классификацию объектов. Это включает в себя изучение алгоритмов машинного обучения и глубокого обучения для распознавания объектов на изображениях.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Сегментация изображений — Текст: электронный — 2011 — URL: <https://habr.com/ru/articles/128768/> (дата обращения: 01.10.2024)
2. **Исрафилов Х.С.** Исследование методов бинаризации изображений / Исрафилов Х.С. – Текст: электронный – Москва, 2018 – URL: <https://scientificjournal.ru/images/PDF/2017/VNO-30/issledovanie-metodov-binarizatsii.pdf> (дата обращения: 10.09.2024)
3. **Вдовин В. А.** Метод адаптивной бинаризации растрового изображения / В. А. Вдовин, А. В. Муравьёв, А. А. Певзне – Текст: электронный – Ярославль, 2012 – URL: <https://vestnik.yspu.org/releases/2012_4e/12.pdf> (дата обращения: 15.10.2024).
4. **Халецкая И. А.** Алгоритмы бинаризации медицинских изображений / И.А. Халецкая, О.Ю. Лысак – Текст: электронный – Томск, 2016 – URL: <https://storage.tusur.ru/files/10961/TU__1204_ALGORITMY_BINARIZATsII_MEDITsINSKIKh_IZO.pdf> (дата обращения: 23.09.2024).
5. Обучение Python GUI (уроки по Tkinter)— Текст: электронный — 2018. — URL: <https://pythonru.com/uroki/obuchenie-python-gui-uroki-po-tkinter> (дата обращения: 26.11.2024).

**ПРИЛОЖЕНИЕ А**

**Основная программа**

from tkinter import \*

from PIL import ImageTk, Image

from tkinter import filedialog

import cv2

import numpy as np

from tkinter import messagebox

import time

root = Tk()

def openfilename():

filename = filedialog.askopenfilename(title='Выбор изображения')

return filename

def program():

messagebox.showinfo(title="О программе", message="""Наименование: Бинаризация изображений

Автор: Агаджанян Алёна Самвеловна

Дата создания: 2024 год""")

def tutorials():

messagebox.showinfo(title="О программе", message="""Для получения бинаризированного изображения вам необходимо нажать \"Выбрать изображение\" во вкладке \"Файл\".

Вы увидите на экране бинаризированные изображения, полученные разными методами.

Вы можете сохранить их по отдельности.""")

def save\_method\_niblack():

if niblack\_imgtk != '':

files = [('JPEG files', '\*.jpeg'),

('PNG Files', '\*.png'),

('Python Files', '\*.py')]

cv2.imwrite(filedialog.asksaveasfilename(title=u'save file ', filetypes=files, defaultextension=files), niblack\_image)

messagebox.showinfo(title="Информация", message="Изображение сохранено")

else:

messagebox.showerror(title="Ошибка", message="Необходимо выбрать изображение")

def save\_method\_bernsen():

if bernsen\_imgtk != '':

files = [('JPEG files', '\*.jpeg'),

('PNG Files', '\*.png'),

('Python Files', '\*.py')]

cv2.imwrite(filedialog.asksaveasfilename(title=u'save file ', filetypes=files, defaultextension=files), bernsen\_imgtk)

messagebox.showinfo(title="Информация", message="Изображение сохранено")

else:

messagebox.showerror(title="Ошибка", message="Необходимо выбрать изображение")

def save\_method\_christian():

if christian\_imgtk != '':

files = [('JPEG files', '\*.jpeg'),

('PNG Files', '\*.png'),

('Python Files', '\*.py')]

cv2.imwrite(filedialog.asksaveasfilename(title=u'save file ', filetypes=files, defaultextension=files), christian\_imgtk)

messagebox.showinfo(title="Информация", message="Изображение сохранено")

else:

messagebox.showerror(title="Ошибка", message="Необходимо выбрать изображение")

def niblack\_thresholding(image, window\_size=15, k=-0.2):

start\_time = time.perf\_counter\_ns ()

# Преобразование изображения в градации серого

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Определяем размеры окна

half\_window = window\_size // 2

# Создаем пустую маску для бинарного изображения

binary\_image = np.zeros\_like(gray)

# Проходим по каждому пикселю изображения

for i in range(half\_window, gray.shape[0] - half\_window):

for j in range(half\_window, gray.shape[1] - half\_window):

# Получаем локальное окно

local\_window = gray[i - half\_window:i + half\_window + 1, j - half\_window:j + half\_window + 1]

# Вычисляем среднее и стандартное отклонение

mean = np.mean(local\_window)

std\_dev = np.std(local\_window)

# Вычисляем порог по методу Ниблэка

threshold = mean + k \* std\_dev

# Применяем порог к текущему пикселю

if gray[i, j] > threshold:

binary\_image[i, j] = 255

timecode = time.perf\_counter\_ns () - start\_time

return binary\_image, timecode/1000000000

def bernsen\_thresholding(image, window\_size=15, delta=10):

start\_time = time.perf\_counter\_ns ()

# Преобразование изображения в градации серого

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Определяем размеры окна

half\_window = window\_size // 2

# Создаем пустую маску для бинарного изображения

binary\_image = np.zeros\_like(gray)

# Проходим по каждому пикселю изображения

for i in range(half\_window, gray.shape[0] - half\_window):

for j in range(half\_window, gray.shape[1] - half\_window):

# Получаем локальное окно

local\_window = gray[i - half\_window:i + half\_window + 1, j - half\_window:j + half\_window + 1]

# Находим минимальное и максимальное значение в окне

min\_val = float(np.min(local\_window))

max\_val = float(np.max(local\_window))

# Вычисляем порог по методу Бернсена

threshold = (min\_val + max\_val) / 2

# Применяем порог с учетом дельты

if gray[i, j] > threshold + delta:

binary\_image[i, j] = 255

elif gray[i, j] < threshold - delta:

binary\_image[i, j] = 0

else:

binary\_image[i, j] = 127 # Полутоны (можно изменить на другой подход)

timecode = time.perf\_counter\_ns () - start\_time

return binary\_image, timecode/1000000000

def christian\_thresholding(image, window\_size=15, c=10):

start\_time = time.perf\_counter\_ns ()

# Преобразуем изображение в оттенки серого

gray\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Вычисляем среднее значение в окне

mean = cv2.boxFilter(gray\_image, ddepth=-1, ksize=(window\_size, window\_size))

# Вычисляем порог по формуле Кристиана

threshold = mean - c

# Бинаризация изображения

binary\_image = (gray\_image > threshold).astype(np.uint8) \* 255

timecode = time.perf\_counter\_ns () - start\_time

return binary\_image, timecode/1000000000

def mse(imageA, imageB):

imageA = np.array(imageA)

imageB = np.array(imageB)

err = np.sum((imageA.astype("float") - imageB.astype("float")) \*\* 2)

err /= float(imageA.shape[0] \* imageA.shape[1])

return err

img = ''

def main():

x = openfilename()

global img, img\_copy

img = Image.open(x)

img\_np = np.array(img)

img\_copy\_gray = cv2.cvtColor(np.array(img), cv2.COLOR\_BGR2GRAY)

img = img.resize((300, 300), Image.Resampling.LANCZOS)

begin\_image = ImageTk.PhotoImage(img)

global begin\_image\_pane

begin\_image\_pane = Label(root, image=begin\_image)

begin\_image\_pane.image = begin\_image

begin\_image\_pane.place(x=30, y=30)

global label\_1

label\_1 = Label(text="Начальное изображение")

label\_1.place(x=100, y=332)

# niblack

global niblack\_image, niblack\_imgtk

niblack\_image, niblack\_time = niblack\_thresholding(img\_np)

niblack\_image\_pil = Image.fromarray(niblack\_image)

niblack\_image\_pil = niblack\_image\_pil.resize((300, 300), Image.Resampling.LANCZOS)

niblack\_imgtk = ImageTk.PhotoImage(niblack\_image\_pil)

global niblack\_pane

niblack\_pane = Label(root, image=niblack\_imgtk)

niblack\_pane.image = niblack\_imgtk

niblack\_pane.place(x=450, y=30)

global label\_2

label\_2 = Label(text="Результат работы метода Ниблэка\nВремя работы метода: " + str(niblack\_time) + " секунд\nMSE: " + str(mse(img\_copy\_gray, niblack\_image)))

label\_2.place(x=500, y=332)

# bernsen

global bernsen\_image, bernsen\_imgtk

bernsen\_image, bernsen\_time = bernsen\_thresholding(img\_np)

bernsen\_image\_pil = Image.fromarray(bernsen\_image)

bernsen\_image\_pil = bernsen\_image\_pil.resize((300, 300), Image.Resampling.LANCZOS)

bernsen\_imgtk = ImageTk.PhotoImage(bernsen\_image\_pil)

global bernsen\_pane

bernsen\_pane = Label(root, image=bernsen\_imgtk)

bernsen\_pane.image = bernsen\_imgtk

bernsen\_pane.place(x=30, y=390)

global label\_3

label\_3 = Label(text="Результат работы метода Бернсена\nВремя работы метода: " + str(bernsen\_time) + " секунд\nMSE: " + str(mse(img\_copy\_gray, bernsen\_image)))

label\_3.place(x=80, y=695)

# christian

global christian\_image, christian\_imgtk

christian\_image, christian\_time = christian\_thresholding(img\_np)

christian\_image\_pil = Image.fromarray(christian\_image)

christian\_image\_pil = christian\_image\_pil.resize((300, 300), Image.Resampling.LANCZOS)

christian\_imgtk = ImageTk.PhotoImage(christian\_image\_pil)

global christian\_pane

christian\_pane = Label(root, image=christian\_imgtk)

christian\_pane.image = christian\_imgtk

christian\_pane.place(x=450, y=390)

global label\_4

label\_4 = Label(text="Результат работы метода Кристиана\nВремя работы метода: " + str(christian\_time) + " секунд\nMSE: " + str(mse(img\_copy\_gray, christian\_image)))

label\_4.place(x=500, y=695)

def delete\_image():

if img!='':

print(img)

begin\_image\_pane.destroy()

niblack\_pane.destroy()

bernsen\_pane.destroy()

christian\_pane.destroy()

label\_1.destroy()

label\_2.destroy()

label\_3.destroy()

label\_4.destroy()

else:

messagebox.showerror(title="Ошибка", message="Необходимо выбрать изображение")

root.title("Binary image")

root.geometry("800x750")

root.resizable()

root.resizable(width=True, height=True)

root.option\_add("\*tearOff", FALSE)

main\_menu = Menu()

file\_menu = Menu()

settings\_menu = Menu()

settings\_menu.add\_command(label="Метод Ниблэка", command=save\_method\_niblack)

settings\_menu.add\_command(label="Метод Бернсена", command=save\_method\_bernsen)

settings\_menu.add\_command(label="Метод Кристиана", command=save\_method\_christian)

file\_menu.add\_command(label="Выбрать изображение", command=main)

file\_menu.add\_command(label="Удалить изображение", command=delete\_image)

file\_menu.add\_cascade(label="Сохранить", menu=settings\_menu)

main\_menu.add\_cascade(label="Файл", menu=file\_menu)

main\_menu.add\_command(label="Помощь", command=tutorials)

main\_menu.add\_command(label="О программе", command=program)

main\_menu.add\_command(label="Выход", command=root.destroy)

root.config(menu=main\_menu)

root.mainloop()